Gaspar-Cunha, A., Ferreira, J. & Recio, G. Evolutionary robustness analysis for multi-objective optimization: benchmark problems. Struct Multidisc Optim 49, 771–793 (2014).

DOI: https://doi.org/10.1007/s00158-013-1010-x

Evolutionary robustness analysis for multi-objective optimization: benchmark problems

António Gaspar-Cunha · Jose Ferreira · Gustavo Recio

Abstract This paper presents a new approach to robustness analysis in multi-objective optimization problems aimed at obtaining the most robust Pareto front solutions and distributing the solutions along the most robust regions of the optimal Pareto set. A new set of test problems accounting for the different types of robustness cases is presented in this study. Non-dominated solutions are classified according to their degree of robustness and are distributed along the Pareto front according to specific algorithm parameter values. Verification of the proposed method is carried out using the developed test problems and artificial and real world benchmark test problems present in the literature.

Keywords Robustness · Multi-objective optimization · Test Problems · Multidisciplinary

1 Introduction

The complexity and multidisciplinary nature of real engineering design and optimization problems makes them difficult to manage within a reasonable time, because they typically have to be described mathematically using sophisticated computational tools that often require significant computational resources. Advances in certain scientific and technological fields (e.g., computational fluid dynamics, structural mechanics), together with the development of more advanced computing techniques (e.g., parallel and/or grid computing) and computer facilities (Bentley 1999) are gradually allowing for more features of complex problems to be addressed.

Many practical engineering problems are multi-objective, i.e., they are characterized by the existence of various, often conflicting, objectives. A practical way to solve such problems is to optimize all objectives simultaneously with the aim of finding the best trade-off between them, i.e., determining how much a given objective is sacrificed to improve others. Multi-Objective Evolutionary Algorithms (MOEAs) are particularly suited for solving such problems because instead of a single solution they use a population of candidate solutions that can be evolved toward the optimal Pareto front (Fonseca and Fleming 1993; Srinivas and Deb 1994; Horn et al. 1994; Knowles and Corne 2000; Gaspar-Cunha and Covas 2004; Deb et al. 2002; Zitzler et al. 2001). This feature allows the obtaining of Pareto frontiers representing the trade-off between the objectives and simultaneously providing a link to the decision variables (Deb 2001; Coello et al. 2006).

Given that in multi-objective optimization the result is a set of solutions showing the trade-off between the objectives, the selection of a single solution to be used in the real problem under study requires information regarding the relative importance of all objectives. This can be done by introducing the preferences of a Decision Maker (DM) in the system. The preference information can be introduced before, during, or after the optimization (Branke et al. 2008; Gaspar-Cunha and Covas 2004; Ferreira et al. 2007). Another important issue, when optimizing real problems is the sensitivity of the solutions obtained when small variations of the design variables or of environmental parameters occur. This indicates that the obtained solutions must be robust, i.e., the performance of the optimal solution(s)

A. Gaspar-Cunha (✉) · J. Ferreira
Institute of Polymers and Composites/I3N, University of Minho, Guimarães, Portugal
e-mail: agc@dep.uminho.pt

G. Recio
Department of Computer Science, Universidad Carlos III de Madrid, Madrid, Spain

should only be slightly affected by them (Jin et al. 2002). There are a considerable number of studies addressing robust optimization (Beyer and Sendhoff 2007). However, robustness is not frequently included in multi-objective optimization algorithms (Ferreira et al. 2008), and qualitative analysis has only recently been proposed (Lee and Kwon 2013).

Generally the application of MOEAs to real engineering problems requires a large number of evaluations of objective functions to obtain an acceptable solution (e.g., on the order of several thousands). Because these evaluations are based on computationally costly methods (such as finite-differences or finite-elements), they are often time-consuming. Consequently, the choice of methods able to reduce the required number of evaluations is of great importance (Jin et al. 2002). A good method consists in the hybridization of MOEAs with local search routines, known as Memetic algorithms (Jin et al. 2002; Gaspar-Cunha and Vieira 2004).

Finally, given the possibility of having a large number of objectives to satisfy simultaneously, it is important to use methodologies that can control the dimensions of the problem. Therefore, it is of great importance to consider ways of reducing the number of objectives, such as using approaches based on statistical techniques (Costa and Oliveira 2007).

In conclusion, the application of a multidisciplinary design optimization methodology to solve multi-objective engineering problems should involve the above questions, i.e., decision making, robustness analysis, reduction of the number of evaluations necessary and selection of the objectives to use. This paper describes a methodology that takes into account robustness for multi-objective optimization, as described next.

As stated above, it is important to consider that, in many cases, the optimal solutions must also be insensitive to perturbations in the design space or perturbations related to environmental parameters, which may arise from insufficient manufacturing accuracy or from noisy design processes, hence, the need for robust multi-objective optimization procedures (Ray 2002). Optimization problems involving such stochastic factors can be typified into four different categories (Jin and Branke 2005): a) when the performance is affected by noise; b) when the values of the design variables change after the optimal solution has been found; c) when the process performance is estimated by an approximation to a real value; d) and when the performance changes with time (dynamic problems). In this work, only the problems of the second category, i.e., those where the design variables change after the optimal solution has been found, will be considered (Ray 2002; Tsutsui and Ghosh 1997).

Two major measures can be used to deal with robustness in an optimization process (Ray 2002; Gaspar-Cunha and Covas 2008; Jin and Sendhoff 2003): expectation measures, where the original objective function is replaced by a measure of both its performance and expectation in the vicinity of the solution considered; and variance measures, where an additional objective, namely, quantifying the deviation around the vicinity of the design point, is created.

Robustness can also be seen as a measure of reliability. In engineering design, a system is considered reliable if it is robust against input and failure uncertainties. On the other hand, a system will have low reliability if a small amount of uncertainty brings about the possibility of failure (Žiha 2000). The work of Damen and Weiland (2002) shows that if the real dynamics of the process change, the performance of the system should not deteriorate beyond an unacceptable level. In practical terms, if a specific process/machine depends, for example, on the definition of a particular temperature, the performance in generating the optimal geometry and/or the operating conditions selected should not deteriorate if the environmental temperature changes (e.g., during winter or summer periods).

In Gunawan and Azarm (2005) the authors present a method for measuring the multi-objective sensitivity of a design alternative and then use it to obtain a set of robust Pareto solutions to a multi-objective optimization problem. Similarly, in Li et al. (2005b) and Li et al. (2009), the authors make use of sensitivity region measures for multi-objective robust, feasibility robust design optimization, problems that have irreducible and reducible interval uncertainty, multiple objective functions and mixed continuous-discrete design variables. Recent approaches to robust optimization focus on using approximation assisted methods. Such is the case in Hu et al. (2012) and Hu et al. (2011) where the authors nicely present new approximation assisted multi-objective robust optimization methods. An integrated design and marketing approach to facilitate the generation of an optimal robust set of design alternatives to carry a product to the prototyping stage was presented in Besharati et al. (2005).

Moreover, a new robustness method proposed in Ferreira et al. (2008) aimed at obtaining the most robust Pareto front solutions and distributing the solutions over the most robust regions of the optimal Pareto set. The method was successfully applied to solve test problems in the literature. However, the problems tested were unable to describe all the possibilities concerning the location of the optimal Pareto front *versus* the location of the most robust front (Deb and Gupta 2006). Thus, further work is needed, which motivated the current research.

The objective of this work is twofold: first, to develop new test problems able to cover different aspects of robustness for multi-objective optimization; second, to modify the previously developed robustness analysis method presented

in Ferreira et al. (2008) to take into account the different types of robustness cases and to study the method's performance over the set of test problems proposed and others available in the the literature (Deb and Gupta 2006). In this study, a detailed investigation will be made of the algorithm parameter values and their effect on the resulting Pareto/robust front solutions. A comparison with the results obtained using the modified NSGA-II will be performed using test problems TP6 to TP9 defined in Section 4 to take into account robustness (Deb and Gupta 2006).

The remainder of the paper is organized as follows. Section 2 presents a comprehensive analysis of robustness concepts. The robust multi-objective evolutionary algorithm will be described in Section 3. Section 4 will give a detailed description of different test problems, either proposed by the authors or obtained from the literature, that will be used to analyze the performance of the robustness algorithms. The experimental results will be presented and discussed in Section 5. Section 6 summarizes the conclusions and main contributions of the work.

2 Robustness analysis

2.1 Concepts

Considering the general minimization problem with one decision variable and a single objective function $f(x)$, a solution x_a is said to be more robust than x_b if the observed magnitude of changes in f under a perturbation δ_x, $\delta_{f_a} = f(x_a + \delta_x)$, is smaller than the observed magnitude of changes in f under the same perturbation for solution x_b, $\delta_{f_a} < \delta_{f_b}$ (Tsutsui and Ghosh 1997). From the above definition, it is clear that the computation of robustness for a given solution involves sampling its neighborhood and using a measure (variance or expectation) that allows taking into account the distribution of objective function values around the original solution in the objective space.

When considering multi-objective optimization problems, various conflicting objectives may co-exist. Thus, a different type of analysis is necessary, for which definitions concerning multi-objective optimization must be introduced. Such problems can be formulated as follows (presented here for minimization problems without loss of generality):

Definition 1 (Multi-objective optimization problem - MOOP)

$$\begin{aligned} \text{minimize}_{x\in\Re^N} \quad & f(x) \\ \text{subject to} \quad & g(x) = 0 \\ & h(x) \leq 0 \\ & x_L \leq x \leq x_U \end{aligned} \tag{1}$$

where, $f(x) = (f_1(x), \ldots, f_M(x)) : \Re^N \to \Re^M$ are the M objective functions of the N decision variables ($x \in \Re^N$), $g(x) = (g_1(x), \ldots, g_J(x)) : \Re^N \to \Re^J$ and $h(x) = (h_1(x), \ldots, h_K(x)) : \Re^N \to \Re^K$ are the J equality and K inequality constraints, respectively, and x_U and x_L are the vectors of the upper and lower bounds of the variables x (i.e., $-\infty \leq x_L^{(i)}, x_U^{(i)} \leq \infty$, for $i = 1, \ldots, N$).

The feasible set, denoted by $\overline{F}$, is the set of points x that satisfy the constraints; that is,

$\overline{F} = \{x \in \Re^N : g(x) = 0,\ h(x) \leq 0,\ x_L \leq x \leq x_U\}$.

The optimal solution of a MOOP is not a single optimal solution but a set of optimal solutions given by all the potential feasible solutions such that the multiple objective functions cannot be simultaneously improved. These solutions are known as Pareto optimal solutions, i.e., the set of non-dominated solutions. Therefore, a solution is optimal, x^*, when it is non-dominated by no feasible solution $x \in \overline{F}$, $x \neq x^*$. In practice it is generally impossible to know the actual optimal set and, consequently, the corresponding Pareto optimal front. Usually, the optimization algorithms find an approximation to this set. The following two definitions apply (Deb 2001):

Definition 2 (Pareto dominance)

Given $x^1,\ x^2 \in \overline{F}$, the point x^1 is said to dominate the point x^2, if

$$\begin{aligned} & f_m(x^1) \leq f_m(x^2), \quad \text{for all} \quad m = 1, \ldots, M \quad \text{and} \\ & f_r(x^1) < f_r(x^2), \quad \text{for at least one r} \quad 1 \leq r \leq M. \end{aligned} \tag{2}$$

Definition 3 (Pareto optimality)

Let $x^* \in \overline{F}$ be a feasible point with the corresponding objective function vector $z^* = f(x^*)$. (x^*, z^*) is Pareto optimal if there is no vector $x \in \overline{F}$, $x \neq x^*$, with

$$\begin{aligned} & f_m(x) \leq f_m(x^*), \quad \text{for all} \quad m = 1, \ldots, M \quad \text{and} \\ & f_r(x) < f_r(x^*), \quad \text{for at least one r} \quad 1 \leq r \leq M. \end{aligned} \tag{3}$$

Definition 4 (Non-dominated set) Among a set of feasible solutions $\mathcal{P}$, the non-dominated set of solutions $\mathcal{P}'$ are those that are not dominated by any member of the set $\mathcal{P}$.

Figure 1 describes the robustness analysis for a problem with two decision variables and two objectives. A local perturbation in the variable space around point A causes a smaller dispersion of the corresponding objective space solutions than when the same amount of local perturbation is considered around solution B. Thus, solution A is considered more robust than solution B. To quantify how robust a solution is, it is necessary to calculate its sensitivity to small perturbations around its neighborhood. For multi-objective problems, the sensitivity must be established with respect to

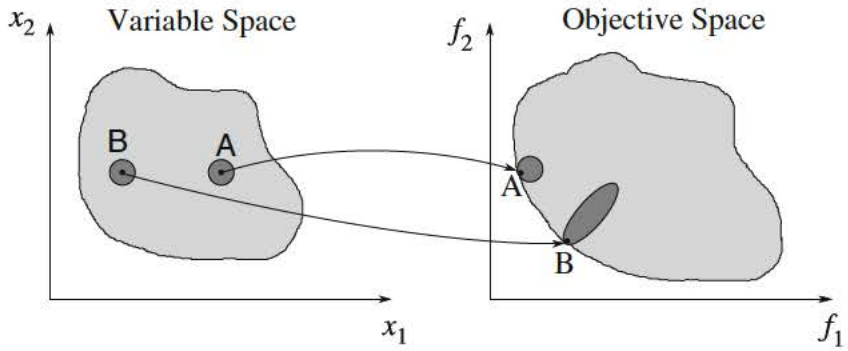


Fig. 1 Concept of robustness for multi-objective optimization problems. Solution A is more robust than solution B

the objectives selected by the decision maker, the combination of which allows the determination of a global measure of sensitivity to decision variable (or variables) perturbation.

2.2 Robustness measures

As introduced in Section 1, robustness can be taken into account by using expectation or variance measures. The former consists of replacing the original objective function by a measure that accounts for both performance and expectation in the vicinity of the solution considered. Several types of expectation measures have been proposed in the literature (Tsutsui and Ghosh 1997; Wiesmann et al. 1998; Jin and Sendhoff 2003; Deb and Gupta 2006; Gaspar-Cunha and Covas 2008). Variance measures involve an additional criterion that measures the deviation of the objective function around the vicinity of the design point. Variance measures only take into account function deviations, ignoring the associated performance. Thus, algorithms using variance measures must consider performance and robustness separately (Gaspar-Cunha and Covas 2008; Jin and Sendhoff 2003; Deb and Gupta 2006).

A recent study in Gaspar-Cunha and Covas (2008) analyzed the performance of selected expectation and variance measures in terms of their capacity to detect robust peaks by assessing features such as the following: i) easy application to problems where the shape of the objective function is not known a priori; ii) the capacity to define robustness regardless of that shape; iii) independence of the algorithm parameters; iv) a clear definition of the function maxima when the fitness *versus* robustness Pareto is represented; and v) efficiency. The best performance was obtained when the following variance measure was used:

$$R_i = \frac{1}{N'} \sum_{j=0}^{N} \frac{\left| \tilde{f}(x_j) - \tilde{f}(x_i) \right|}{|x_j - x_i|}, \qquad d_{i,j} < d_{max} \tag{4}$$

where the robustness of individual i is defined as the average value of the ratio of the difference between the normalized fitness of individual i, $\tilde{f}(x_i)$, and that of its neighbors (j), over the distance separating them. In the above expression, $\tilde{f}(x_i) = (f(x_i) - f_{min})/(f_{max} - f_{min})$ is used for the maximization and $\tilde{f}(x_i) = 1 - (f(x_i) - f_{min})/(f_{max} - f_{min})$ is for the minimization of the objective function $f(x_i)$, where f_{max} and f_{min} represent the limits of the objective function variation, and N' is the number of population individuals whose Euclidean distance between points i and j is lower than a given threshold ($d_{i,j} < d_{max}$). The limits for each objective function are defined by the user. Only variance measures will be used in this work.

Multi-objective robust optimization algorithms aim at obtaining a set of solutions (not necessarily optimal Pareto solutions, as will be explained below) that are both the best approximation to the optimal Pareto set and robust. Four main situations can be described that arise from real optimization problems in which the location of the optimal Pareto front versus the location of the robust region and/or optimal Pareto robust front. These situations, correspond to each of the plots shown in Fig. 2 (Deb and Gupta 2006; Gunawan and Azarm 2005; Ferreira et al. 2008). In type 1 problems (Fig. 2a) all solutions on the Pareto optimal frontier are robust. In type 2 problems (Fig. 2b) the solutions are characterized by the fact that only some of the solutions belonging to the Pareto optimal frontier are robust. In type 3 problems, represented in Fig. 2c, the Pareto front solutions are not robust; instead, a different robust Pareto frontier exists. Finally, Fig. 2d represents problems of type 4, in which some of the robust solutions belong to the optimal Pareto frontier but others do not. The test problems studied in Section 4 can be classified as belonging to one of these four categories.

2.3 Robustness sampling

In robustness analysis, there are a number of ways of generating neighboring points in the vicinity of a solution (Branke 2000; Saha and Ray 2011). The simplest strategy may be to randomly create the points around the solution. However, this approach introduces randomness in evaluating the same solution more than once. In Branke (1998), the use of a random pattern, which is repeated in consecutive iterations, was suggested. There are also several statistical space

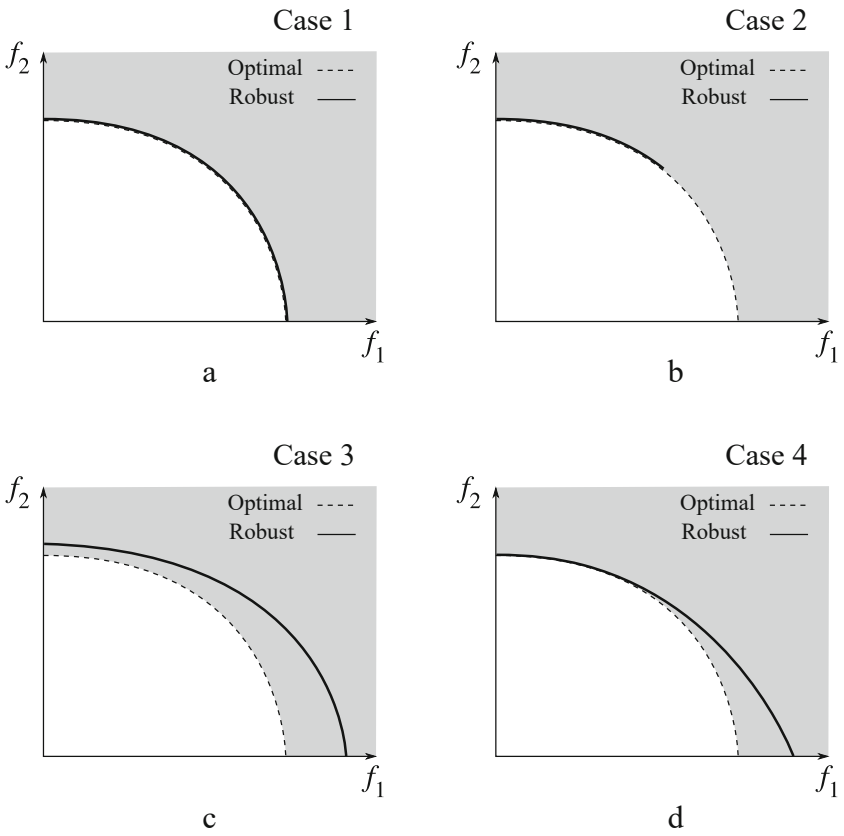


Fig. 2 Optimal Pareto frontier versus robust Pareto frontier for convex minimization example problems (grey areas represent dominated regions). Test problems can be classified into one of the above four cases when comparing the optimal Pareto front with the robust Pareto front

filling methods for creating a well distributed and diverse set of points in a specific region (Montgomery 2001). In Saha and Ray (2011), a partial scheme was employed to generate the neighbors. A common strategy known as the Latin hypercube consists in dividing the perturbation domain into a grid and sampling the neighbors in such a way that all small hyper-boxes within the grid are represented (Montgomery 2001; Deb and Gupta 2006). A illustrative example of this strategy is shown in Fig. 3, where the vicinity of a given solution (a, b) is divided into a 6x6 grid. The neighbors that will contribute to the computation of robustness for solution (a, b) will be distributed across the grid in such a way that there cannot be two neighbors in the same row or column of the grid, as shown in the left of Fig. 3. For some problems, it might be necessary to analyze the robustness behavior only concerning variations around a single (or more, but not all) decision variable (or direction of search), i.e., it is assumed that the robustness of the solutions will not depend on changes to all existing decision variables. This is the case in the example cited above, in which a process only

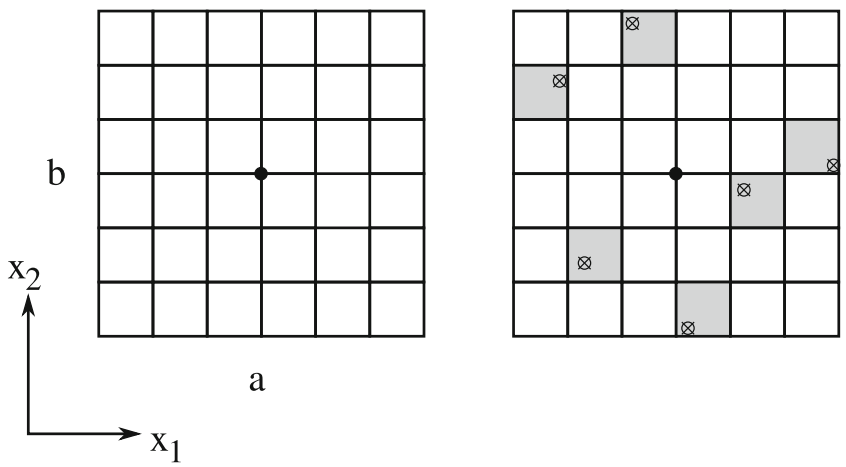


Fig. 3 Graphical example of creating six neighboring points around a solution $x = (a, b)$ using the Latin hypercube method. The neighborhood is divided into a grid and then the six neighbors are taken at a random location for each grid cell

depends on changes occurring in the environment temperature. This will indicate how sensitive the solutions are to changes in a particular decision variable.

Different approaches for generating neighbors will be tested in Section 5: i) In the first approach, a single direction x_i will be used to generate the neighbors. In the test problems presented in Section 4, this direction is well defined. ii) In the second case, the neighbors (x_i') will be generated as a random number in the interval $[x_i - \Delta p; x_i + \Delta p]$, were Δp is a parameter indicating the minimum distance between the neighbors, and x_i the point considered. Δp is applied individually for each design variable. iii) In the third alternative, the neighbors will be generated randomly but proportionally to the interval between the maximum and minimum of each decision variable. This is performed as follows: x_i' is equal to a random number in the interval $[x_i - \Delta p * |max(x_i) - min(x_i)|; x_i + \Delta p * |max(x_i) - min(x_i)|]$. iv) Finally, the Latin hypercube method was used. In Section 5 the influence of the Δp parameter on the algorithm performance will be studied in detail for all cases.

3 Introducing robustness in MOEAs

3.1 Multi-objective algorithm

Multi-Objective Optimization Algorithms (MOOAs) must accomplish two basic functions simultaneously. First, they need to guide the population toward the optimal Pareto set. This can be done by using a fitness assignment operator that takes into account the non-dominance concept. Then, the non-dominated set must be kept as diverse as possible, i.e., the solutions must be well distributed in the entire optimal Pareto front. This is usually done through the use of a density estimation operator, such as fitness sharing (Goldberg and Richardson 1987; Deb and Goldberg 1989) and the crowding distance operator (Deb et al. 2002). It is also necessary to maintain an archive of the best solutions found during the various generations to prevent some non-dominated solutions from being lost (Knowles and Corne 2003). In this case, an elitist population of the best individuals is kept in an archive. In MOEAs, therefore, it is generally only necessary to replace the selection phase of a traditional EA by a routine able to deal with multiple objectives (Deb 2001).

The MOEA adopted in this work is the Reduced Pareto Set Genetic Algorithm (RPSGA) (Gaspar-Cunha and Covas 2004). The main steps of this algorithm are illustrated in Algorithm 1. Initially, an internal population of size N is randomly defined, and an empty external population of size $2N$ is created. At each generation, i.e., when a stop condition is not met, the following operations are performed: i) the internal population is evaluated using the modeling routine; ii) a clustering technique is applied to reduce the number of solutions on the efficient frontier and to calculate the ranking of the individuals of the internal population; iii) the fitness of the individuals is calculated using a ranking function; iv) a fixed number of best individuals are copied to the external population; v) if the external population is not completely full, the genetic operators of selection, crossover and mutation are applied to the internal population to generate a new population; and vi) when the external population becomes full, the clustering technique is applied to sort the individuals of the external population, and a pre-defined number of the best individuals are incorporated in the internal population by replacing the individuals with the lowest fitness.

Detailed information about this algorithm can be found in Gaspar-Cunha (2009), Gaspar-Cunha and Covas (2004). The influence of some important parameters of the algorithm, such as the size of the internal and external populations, the number of individuals copied to the external population in each generation and from the external to internal population and the limits of the indifference of the clustering technique, have already been studied and the best values selected (Gaspar-Cunha and Covas 2004).

Algorithm 1 Reduced Pareto set genetic algorithm (RPSGA)

```
Create a random initial population (internal);
Create an empty external population;
while Not Stopping Condition do
    Evaluate the internal population;
    Compute the ranking of individuals using clustering;
    Compute the fitness of the individuals using a ranking
    function;
    Copy the best individuals to the external population;
    if External population becomes full then
      Apply the clustering to this population;
      Copy the best individuals to the internal population;
    end if
    Select the individuals for reproduction;
    Crossover;
    Mutation;
    Archive best solutions;
end while
```

3.2 Robustness calculation methodology

The methodology proposed in this section for taking into consideration robustness in MOEAs can be applied using any multi-objective algorithm without having to make considerable changes. For the purpose of this work, the RPSGA algorithm presented in the previous section will be used.

Usually, only three additional steps are necessary: the first is needed to calculate the niche count using sharing functions (Deb and Goldberg 1989), the second to calculate the robustness of the individuals in the population and the third to include robustness in the global fitness function calculation. The presented methodology consists of modifications of the previous methodology proposed by the authors in Ferreira et al. (2008). These modifications were introduced to take into account the characteristics of the different types of robustness scenarios shown in Fig. 2.

The calculation of robustness depends strongly on the neighboring points used to calculate it. Thus, the aspects referred to in Section 1 must be considered, i.e., the possibility of selecting the number of points, the possibility of choosing the method for defining those points (single direction, random, random proportional and Latin hypercube) and the possibility of selecting one or more directions of search in the decision space domain. In all these cases, the following equation, representing a variance measure similar to (4), is used to calculate robustness (R_i) on point i:

$$R_i = \frac{1}{N'} \sum_{j=0}^{N'} \frac{\left| f(x_j) - f(x_i) \right|}{\left| x_j - x_i \right|}, \qquad d_{i,j} < d_{max} \tag{5}$$

where N' represents the number of neighbors, j, of i such that the distance ($d_{i,j}$) between the points is not greater than d_{max}. The distance can be calculated as:

$$d_{i,j} = \sqrt{\sum_{k=1}^{L} \left(x_{k,j} - x_{k,i} \right)^2} \tag{6}$$

An important characteristic of the proposed method is the possibility for a Decision Maker (DM) to define an ϵ value ranging in the interval [0, 1] that quantifies the degree of robustness to be included in the search (Ferreira et al. 2008). This parameter indicates the level of dispersion desired in the final results. If a small value is chosen (near 0), the optimal Pareto frontier only takes into account the most robust solutions, while if a high value is selected (near 1) the entire optimal Pareto frontier can be obtained (depending on the problem). Therefore, if the user has good knowledge of the process and about the relative importance of the various objectives (defined through the weights), a small ϵ value can be selected. In the opposite case, the optimization process can start with a higher ϵ value and the optimization algorithm can run during some generations. Then, after analyzing the results, the user can define a smaller ϵ, and the process can continue using the population obtained in the previous step. To accomplish this goal, the ϵ value (which varies linearly) defined by the DM must be transformed as follows:

$$\epsilon' = (\gamma - \delta) \left[\frac{1}{exp\left[(1-\epsilon) \right]^3} \right]^8 + \delta \tag{7}$$

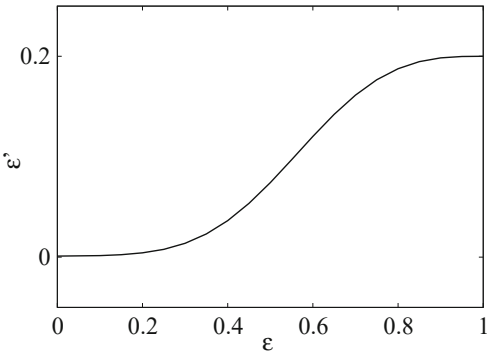


Fig. 4 ϵ' as a function of ϵ ($\gamma = 0.2$ and $\delta = 0.001$)

where γ and δ are values to be determined experimentally, γ defines the maximum value of ϵ' and δ is the minimum value. In the present study the best values obtained for these parameters were 0.2 and 0.001 for γ and δ, respectively. Figure 4 shows ϵ' as a function of ϵ in this case. For small and high ϵ values, ϵ' grows slowly, while for intermediate values the variation is nearly linear. The aim was to achieve a balance for the contributions to the global fitness of the solutions (non-dominance, dispersion of the solutions and robustness).

The diversity of the solutions is maintained using the sharing function as defined by Goldberg and Richardson (1987):

$$Sh(d_{i,j}) = \begin{cases} 1 - \left(\frac{d_{i,j}}{\sigma_{share}} \right)^2 & , if\, d_{i,j} < \sigma_{share} \\ 0 & \text{otherwise} \end{cases}. \tag{8}$$

where σ_{share} is a constant that must be determined experimentally. Because the basic idea of sharing is to deteriorate the fitness when the number of neighbors is larger, the final fitness must be divided by the niche count (m_i), calculated as follows:

$$m_i = \sum_{j=1}^{N} sh(d_{i,j}) \tag{9}$$

The global fitness of the population individuals (F_i) can now be determined by taking into account the following: a) the non-dominance ranking ($Rank_i$) calculated using the clustering technique referred to above; b) the niche count (m_i), computed by (8) and (9); and the robustness obtained by (5). The contribution of each of these three values for the final fitness is balanced through the ϵ' value, as shown in the following equation:

$$F_i = \frac{1}{Rank_i} + \left[\frac{1}{1-\epsilon'} - 1 \right] \frac{m_i}{m_i + 1} + (1-\epsilon') \frac{R_i}{R_i + 1} \tag{10}$$

where R_i represents a robustness variance measure. Globally, however, the concepts of variance and expectation are mixed in this final fitness equation. In (10), the final value of F_i is also balanced by the value of ϵ'. When ϵ' approaches

0.2 (i.e., when $\epsilon = 1$, see Fig. 4), the contribution of the robustness (R_i) is small and that of the niche count (m_i) has the maximum value. The opposite is also true, i.e., when ϵ' approaches zero, the contribution of the robustness to F_i has the maximum value.

Figure 5 illustrates the contribution of the non/dominance ranking ($Rank_i$), niche count (m_i) and robustness (R_i). An analysis of these graphs shows that the contribution of the niche count is very small, but it is necessary and sufficient to help the MOEA better distribute the solutions along the optimal Pareto front. Also, as desired, the fitness deteriorates with an increasing ranking value and niche count.

The computations for taking into account robustness in MOEA are illustrated in Algorithm 2, which is a modification of Algorithm 1. Here, the step to compute the fitness using a ranking function is replaced by three new steps in Algorithm 2. These steps are the following: i) compute the niche count of individuals using sharing functions; ii) compute the robustness of individuals using (5); and iii) compute the fitness of individuals using (10).

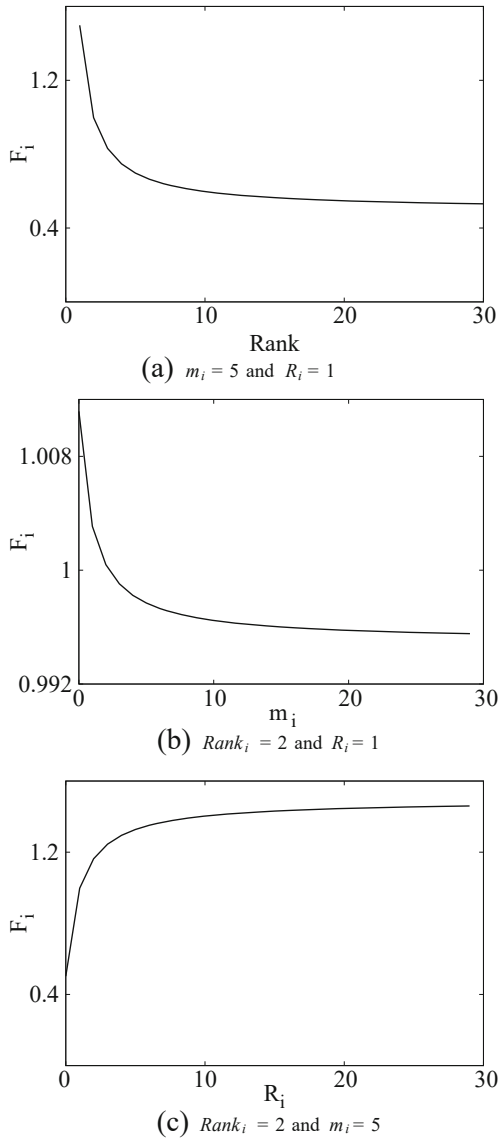


Fig. 5 Contribution to the global fitness of: **a** non-dominance ranking; **b** niche count; **c** robustness

Algorithm 2 RPSGA with robustness

Create random initial population (internal);
Empty external population;
while Not Stopping Condition **do**
  Evaluate internal population;
  Compute the ranking of individuals using clustering;
  Compute the niche count of the individuals using sharing functions;
  Compute the robustness of individuals using (5)**;**
  Compute the fitness of the individuals using (10)**;**
  Copy the best individuals to the external population;
  if External population becomes full **then**
    Apply the clustering to this population;
    Copy the best individuals to the internal population;
  end if
  Select the individuals for reproduction;
  Crossover;
  Mutation;
  Archive best solutions;
end while

4 Test problems

The aim of this section is to propose and describe test problems to validate the robustness approach proposed and to serve as benchmark test problems. These test problems must be able to cover different aspects of the relation between the optimal Pareto front and the most robust region in the feasible search space, i.e., the type of situations described in Fig. 2. Simultaneously, convergence to the optimal Pareto front must be challenging to the MOEAs, which can be accomplished by using a large number of decision variables, convex and non-convex fronts, discontinuous Pareto fronts and multi-modal problems. Due to the difficulty of finding test problems in the literature that are able to cover all the aspects stated above, test problems TP1 to TP5 are proposed here. Therefore, for the purpose of this work, the following problem types were defined or taken from the literature:

1. Test problems where the entire optimal Pareto front is robust, as in Test Problems 2 (TP2) and 6 (TP6) (Deb and Gupta 2006).
2. Test problems where only a partial portion of the optimal front is robust. TP1 has the most robust region

on the convex section of the optimal Pareto front, and TP3 has the most robust region located in a non-convex optimal Pareto front.

3. A test problem with discontinuous robust regions, with the robustness of these regions differing from each other. The aim in this case is to assess the ability of the algorithm to converge to the robust region (these are the characteristics of TP4).
4. A test problem with discontinuous robust regions, but now with equal robustness. The aim is to assess the ability of the algorithm to converge to distinct sections of the Pareto front (TP5).
5. Test problems where a local optimal front exits (i.e., multi-modal problems) and where the entire local optimal front or only part of it is robust (TP7 to TP9) (Deb and Gupta 2006).
6. A real test problem (TP10) (Deb and Gupta 2006).

Test problems TP1 to TP5 were developed for the purpose of this paper, while test problems TP6 to TP10 were obtained from the literature. TP1 to TP5 are presented in detail in Table 1. The optimal Pareto front and the evaluation of the robustness along this front were plotted to gain a better understanding of the behavior of each test problem. In this table, lower values represent higher robustness. The optimal Pareto front corresponds to $x_i = 0$ for $i = 2, 3, ..., L$ and x_1, ranging in the interval [0, 1]. This means that when the solution approaches the optimal Pareto front, it only makes sense to calculate the robustness for changes in the decision variable x_1, as the others must be all equal to 0. Therefore, the robustness plot shown in Table 1 is represented as a function of f_1 (equal to x_1). This characteristic and its influence on the algorithm performance will be discussed in the results section.

Test problems 6 to 9 (TP6 to TP9) were used in Deb and Gupta (2006). The aim here is to take into account a different type of behavior and to compare the results obtained by Deb and Gupta (2006) with those produced using the methodology proposed here. These are multi-modal problems where the most robust region of the feasible objective space belongs to a local Pareto front and not to the global Pareto front (except for the case of TP6). Because MOEAs have great difficulty converging to the global Pareto front on these type of problems, it is important to test the ability of the algorithm to converge to the robust front.

Test problem 6 (TP6) illustrates a scenario where the complete original efficient front is robust (Deb and Gupta 2006).

$$
\begin{aligned}
&\text{Minimize} \quad f_1(\mathbf{x}) = x_1,\\
&\text{Minimize} \quad f_2(\mathbf{x}) = h(x_1) + g(\mathbf{x})S(x_1),\\
&\text{Subject to} \quad 0 \le x_1 \le 1, -1 \le x_i \le 1, \quad i = 2, 3, ..., n,\\
&\text{where} \quad h(x_1) = 1 - x_1^2,\\
&\qquad g(\mathbf{x}) = \sum_{i=2}^{L} 10 + x_i^2 - 10\cos(4\pi x_i),\\
&\qquad S(x_1) = \frac{\alpha}{0.2 + x_1} + \beta x_1^2.
\end{aligned} \tag{11}
$$

The parameters suggested are $\alpha = 1$ and $\beta = 1$. The efficient Pareto front corresponds to $x_i = 0$ for $i = 2, 3, ..., L$ and for any value of x_i in the prescribed domain [0, 1]. Therefore, the following relationship between objectives is obtained:

$$f_2 = 1 - f_1^2 \tag{12}$$

which is graphically described in Fig. 6a.

A variation of the above problem, which will be named Test Problem 7 (TP7), represents a situation where a part of the original efficient front is no longer robust (Deb and Gupta 2006). The formulation for this problem is the same as for TP6, except for the value of the β parameter, which is now equal to 10 ($\beta = 10$). The change made in β confirms that the right side of the front shown in Fig. 6a is less robust than the left side. The parameters suggested for this problem are $\alpha = 1$ and $\beta = 10$.

In test problem 8 (TP8), the original optimal Pareto front is less robust than a local front, as shown in Fig. 6b. This is a bi-modal and two-objective optimization problem that requires "swapping" to the local robust front (Deb and Gupta 2006).

$$
\begin{aligned}
&\text{Minimise} \quad f_1(\mathbf{x}) = x_1,\\
&\text{Minimise} \quad f_2(\mathbf{x}) = h(x_2)(g(\mathbf{x}) + S(x_1)),\\
&\text{Subject to} \quad 0 \le x_1, x_2 \le 1, -1 \le x_i \le 1, \quad i = 3, 4, ..., n,\\
&\qquad \text{where} \quad h(x_2) = 2 - 0.8\exp\left(-\left(\frac{x_2 - 0.35}{0.25}\right)^2\right)\\
&\qquad\qquad - \exp\left(-\left(\frac{x_2 - 0.85}{0.03}\right)^2\right),\\
&\qquad\qquad g(\mathbf{x}) = \sum_{i=3}^{L} 50x_i^2,\\
&\qquad\qquad S(x_1) = 1 - \sqrt{x_1}.
\end{aligned} \tag{13}
$$

Both local and global efficient fronts correspond to $x_i = 0$ for all $i = 3, 4, ..., L$, resulting in the following relationship between objectives for local and global fronts, as shown in Fig. 6b.

$$
\begin{aligned}
f_2 &= 1 - \sqrt{f_1}(\text{Global})\\
f_2 &= 1.2(1 - \sqrt{f_1})(\text{Local})
\end{aligned} \tag{14}
$$

Test problem 9 (TP9), shown in Fig. 6c, represents a scenario where a part of the optimal Pareto front is robust

Table 1 Test Problems 1 to 5 (TP1 to TP5). Right: equations and parameters; Left: optimal Pareto fronts and robustness

Test Problem 1 (TP1)

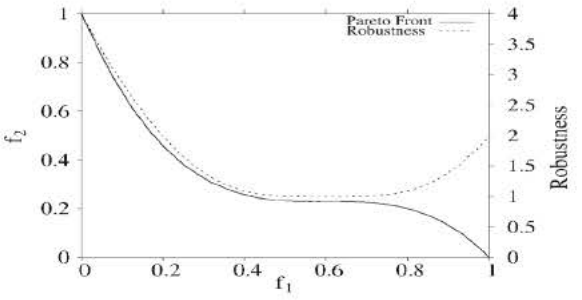
minimize $f_1(\mathbf{x}) = x_1$
minimize $f_2(\mathbf{x}) = h(x_1) + g(\mathbf{x})s(x_1)$
subject to: $0 \leq x_i \leq 1, \quad i = 1, ..., L$
where:

$$h(x_1) = \frac{(x_1 - 0.6)^3 - 0.4^3}{-0.6^3 - 0.4^3}$$

$$g(\mathbf{x}) = \left(\frac{\sum_{i=2}^{L} x_i}{L-1}\right)^2$$

$$s(x_1) = \frac{1}{x_1 + 0.2}$$

front solutions correspond to
$x_i = 0 \quad$ for $i > 1$

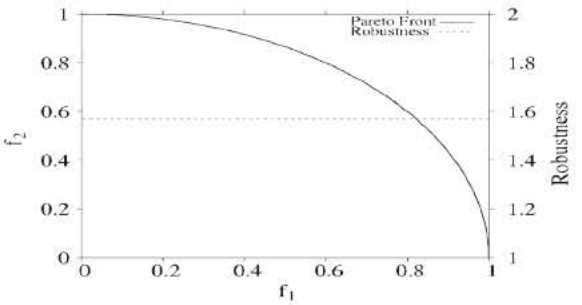


Test Problem 2 (TP2)

minimize $f_1(\mathbf{x}) = cos\left(\frac{\pi x_1}{2}\right)$
minimize $f_2(\mathbf{x}) = g(\mathbf{x}) sin\left(\frac{\pi x_1}{2}\right)$
subject to: $0 \leq x_i \leq 1, \quad i = 1, ..., L$
where:

$$g(\mathbf{x}) = 1 + 10\frac{\sum_{i=2}^{L} x_i}{L-1}$$

front solutions correspond to
$x_i = 0 \quad$ for $i > 1$

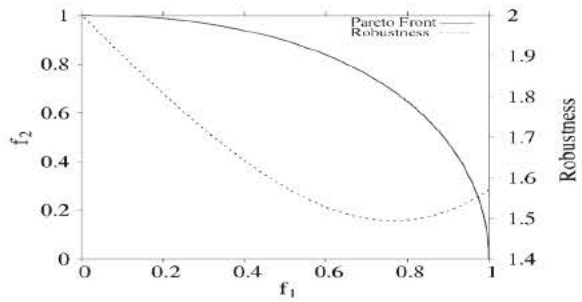


Test Problem 3 (TP3)

minimize $f_1(\mathbf{x}) = 1 - x_1^2$
minimize $f_2(\mathbf{x}) = g(\mathbf{x}) sin\left(\frac{\pi x_1}{2}\right)$
subject to: $0 \leq x_i \leq 1, \quad i = 1, ..., L$
where:

$$g(\mathbf{x}) = 1 + 10\frac{\sum_{i=2}^{L} x_i}{L-1}$$

front solutions correspond to
$x_i = 0 \quad$ for $i > 1$

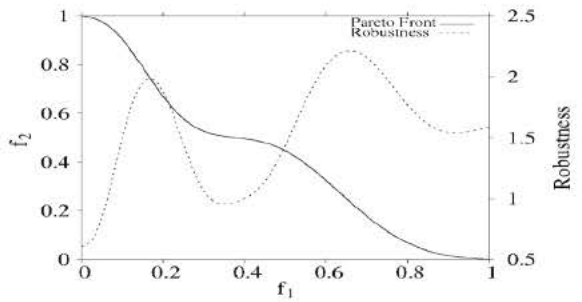


Test Problem 4 (TP4)

minimize $f_1(\mathbf{x}) = \left(\frac{e^{x_1} - 1}{e - 1}\right)$
minimize $f_2(\mathbf{x}) = g(\mathbf{x})\left[\frac{sin(4\pi x_1) - 15x_1}{15} + 1\right]$
subject to: $0 \leq x_i \leq 1, \quad i = 1, ..., L$
where:

$$g(\mathbf{x}) = 1 + 10\frac{\sum_{i=2}^{L} x_i}{L-1}$$

front solutions correspond to
$x_i = 0 \quad$ for $i > 1$

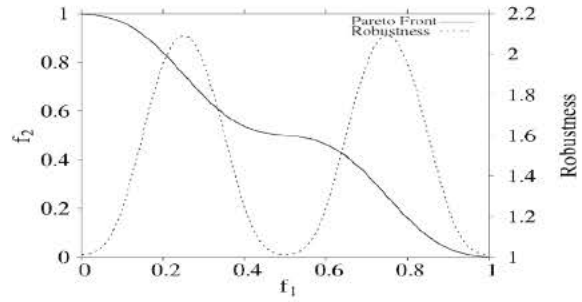


Test Problem 5 (TP5)

minimize $f_1(\mathbf{x}) = x_1$
minimize $f_2(\mathbf{x}) = g(\mathbf{x})\left[\frac{sin(4\pi x_1) - 15x_1}{15} + 1\right]$
subject to: $0 \leq x_i \leq 1, \quad i = 1, ..., L$
where:

$$g(\mathbf{x}) = 1 + 10\frac{\sum_{i=2}^{L} x_i}{L-1}$$

front solutions correspond to
$x_i = 0 \quad$ for $i > 1$



together with a part of a local efficient front (Deb and Gupta 2006). The formulation is similar to that of TP8 with some modifications. The bound variable of x_2 changes, $-0.15 < x_2 < 1$, and the function $h(X)$ is now calculated as follows:

$$h(x_1, x_2) = 2 - x_1 - 0.8 \exp\left(-\left(\frac{x_1 + x_2 - 0.35}{0.25}\right)^2\right) - \exp\left(-\left(\frac{x_1 + x_2 - 0.85}{0.03}\right)^2\right) \quad (15)$$

This problem has it global efficient front around $x_1 + x_2 = 0.85$ and the local efficient front around $x_1 + x_2 = 0.35$. The number of decision variables used, in the cases of TP6 to TP9, are $L = 5$.

Test problem 10 (TP10) is commonly used in the engineering design optimization literature, for example, in Li et al. (2005a), Kirsch (1981), Deb (2001) and Barrico et al. (2009). The resulting Pareto front is represented in Fig. 6d. It consists of a real problem where the governing equations come from the design of a two-bar truss subject to a single vertical load of 100kN, applied at the end of the beam. For further details see Deb (2001).

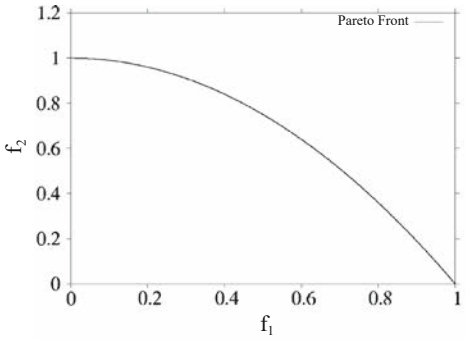
$$\begin{aligned}
&\text{Minimise} && f_1(\mathbf{x}) = x_1\sqrt{16 + x_3^2} + x_2\sqrt{1 + x_3^2},\\
&\text{Minimise} && f_2(\mathbf{x}) = \frac{20\sqrt{16 + x_3^2}}{x_1 x_3},\\
&\text{Subject to} && 20\sqrt{16 + x_3^2} - 100x_3x_1 \le 0,\\
& && 80\sqrt{1 + x_3^2} - 100x_3x_2 \le 0,\\
& && x_1 > 0,\\
& && x_2 \ge 0,\\
& && 1 \le x_3 \le 3. \qquad (16)
\end{aligned}$$

5 Results and discussion

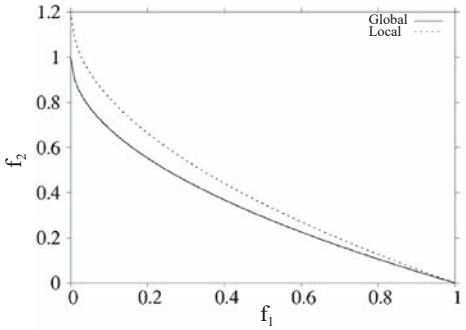
5.1 Experimental set up

Three type of studies will be performed. First, the influence of algorithm parameters will be studied using test problems TP1 to TP5, with the aim of assessing the ability of the robustness methodology proposed to obtain robust solutions and to determine the best parameters to use. Then, a comparison with the results of Deb and Gupta (2006) will be made using test problems TP6 to TP9. Finally, the proposed approach will be tested using a real problem from engineering (TP10).

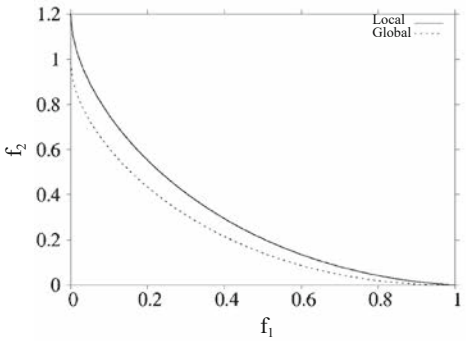
The following RPSGA parameters were used (see Gaspar-Cunha and Covas (2004) for more details): the main



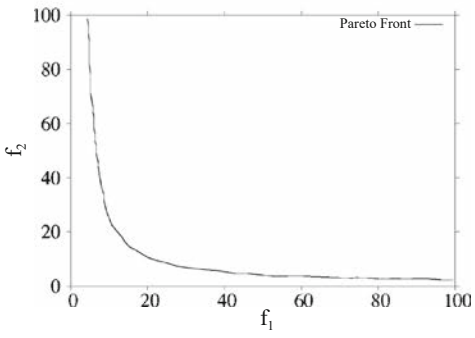
(a) Test Problems 6 and 7 (TP6 and TP7)



(b) Test Problem 8 (TP8)



(c) Test Problem 9 (TP9)



(d) Test Problem 10 (TP10)

Fig. 6 Optimal Pareto and robust fronts for test problems 6 to 10 (TP6 to TP10)

Table 2 Computational runs for testing the influence of the algorithm parameters

Case Study	ϵ[a]	σ_{sh}	Δp	N′	Type	Seed
Case 1	0.03, 0.1, 0.2, 0.3, 0.5, 0.8, 1.0	0.1	0.1	30	x_1	600
Case 2	0.2	0.007, 0.01 0.05, 0.1, 0.2	0.1	30	x_1	600
Case 3	0.2	0.1	0.007, 0.01 0.05, 0.1, 0.2	30	x_1	600
Case 4	0.2	0.1	0.1	10, 30, 50	x_1	600
Case 5	0.2	1.0	0.05	30	**b**	600
Case 6	0.2	1.0	0.05	30	x_1	350
						600
						1100

[a] in the cases of TP4 and TP5, the ϵ values used are 0.02, 0.3, 0.4, 0.6, 0.8 and 1.0; [b] four different types of sampling the neighbors are used: direction x_1, random, random proportional and Latin hypercube

and elitist populations consist of 100 and 200 individuals, respectively; a roulette wheel selection strategy was adopted; and, for the genetic operators of the algorithm, a crossover probability of 0.8, a mutation probability of 0.05, 30 ranks and a 0.01 limit of indifference of the clustering technique. The total number of generations used in the experiments were 2000 for TP1 to TP5, 5000 for TP6 to TP9 and 500 for TP10.

5.2 Influence of algorithm parameters

Table 2 shows the set of computational runs made for testing the influence of the algorithm parameters for test problems TP1 to TP5. These parameters are the following: i) a dispersion parameter (ϵ); ii) the maximum distance between solutions, i.e., σ_{sh} in (8) iii); the minimum distance between the neighboring points selected (Δp); iv) the number of neighbors (N'); v) the type of method to calculate robustness, which includes the use of a single direction, random selection, random proportional and Latin hypercube; and vi) the seed value of the random numbers to test the convergence dependency of the initial population.

Figures 7, 8, 9, 10, and 11 show the influence of ϵ on the size and distribution of the robust Pareto front obtained for TP1 to TP5, respectively (see also Table 1). As expected, the size of the Pareto front obtained increases with ϵ. In the case of TP1 (Fig. 7), for large values of ϵ (> 0.5) some solutions around the most robust region were lost, as there is a plateau where the robustness is constant, as can be seen in Table 1. Consequently, the balance between the factors used in Fitness (10) causes the algorithm to converge to the solutions around point $f_1 = 0.6$, where the robustness is higher because less importance is attributed to robustness. In (10), the term including robustness is multiplied by $(1-\epsilon')$. Thus, if ϵ' increases, the importance of robustness decreases. TP2 (Fig. 8) is a particular case where the robustness is equal in the entire domain, and therefore, for small ϵ values the algorithm converges to the central region.

In the case of TP3, the behavior is identical to that of TP1, i.e., for small ϵ values the algorithm converges only to the robust region. TP4 and TP5 have discontinuous robust regions.The difference is that in the case of TP5 these regions have different robustness values. The algorithm is able to detect this characteristic and, in the case of TP4 for small ϵ values, it converges to the most robust regions, while in the case of TP5 it converges simultaneously to the discontinuous regions, which have equal robustness. Figure 12 shows the influence of the number of neighbors (N'), and, as can be seen, the effect is small. Due to lack of space, the other results will not be presented here. The remaining results can be found in the supplementary results web page[1].

Table 3 presents a summary of the influence of the algorithm parameters studied concerning test problems TP1 to TP5. The most important conclusion that can be drawn from this table is that the ϵ parameter works well in all test problems and that the other parameters have little influence on the algorithm's performance. An exception is the case of TP2, where the best values to use for σ_{sh} and δ_p are 0.1 for both variables. Also, in all cases, the best performance is attained when the way in which the neighbors are sampled takes into consideration only the x_1 direction. Also, as expected, the best type of neighbor sampling is the one in

[1] http://www.dep.uminho.pt/agc/agc/Supplementary_Information_Page.html

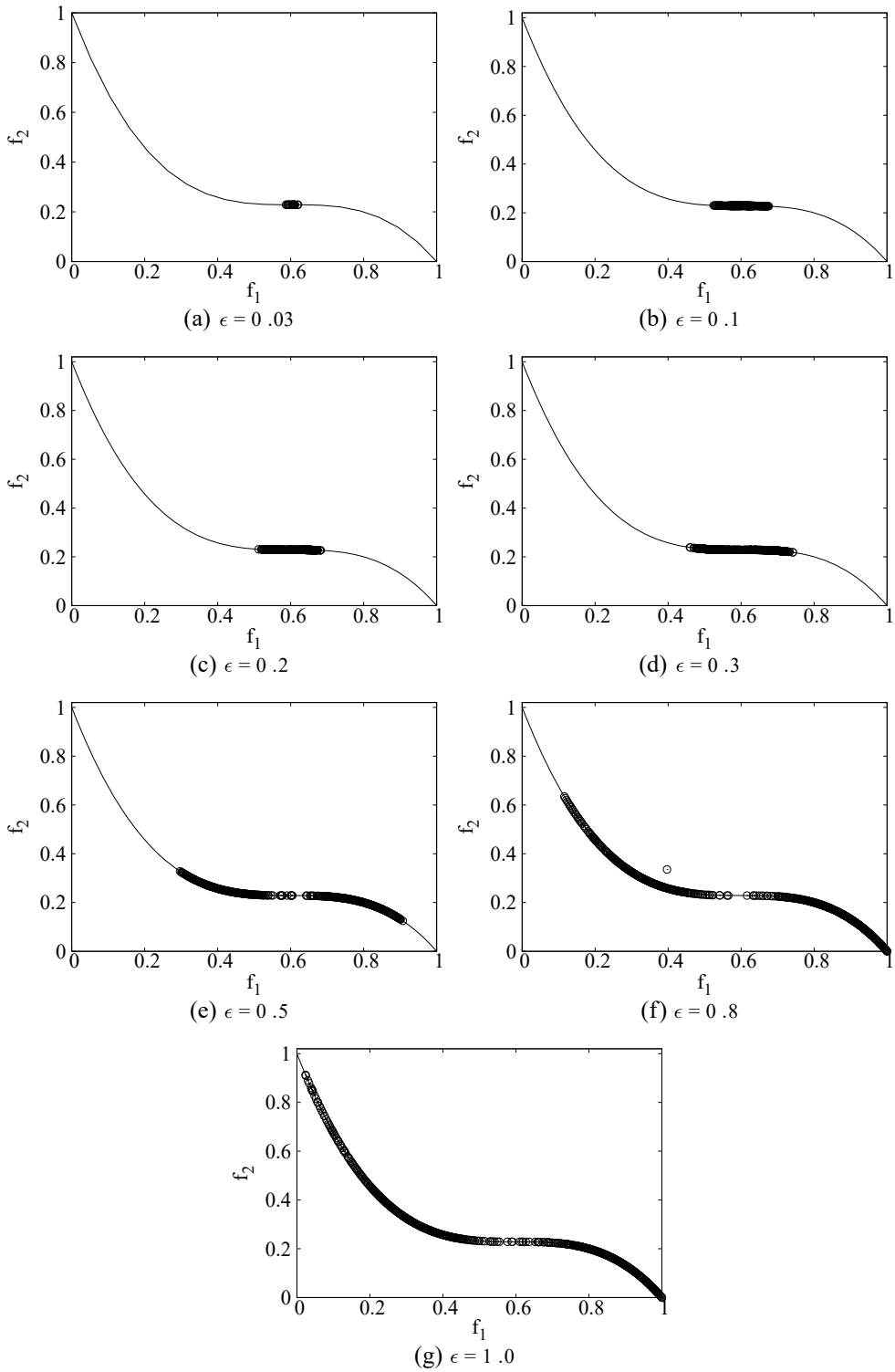


Fig. 7 Influence of the dispersion parameter, ϵ, on the size of the robust region obtained for TP1 (continuous line: Pareto front; points: solutions obtained)

which only x_1 varies, as the problems were developed with this characteristic. Therefore, the recommended values are as follows: ϵ depends on the desired extent of the Pareto front to be obtained and/or the degree of certainty that the Decision Maker (DM) has, $\sigma_{sh} = 0.1$, $\Delta_p = 0.1$, $N' = 30$ and the method used to calculate robustness, direction x_1.

These results are entirely new and very useful, especially when the DM has to select a robust solution (or a small

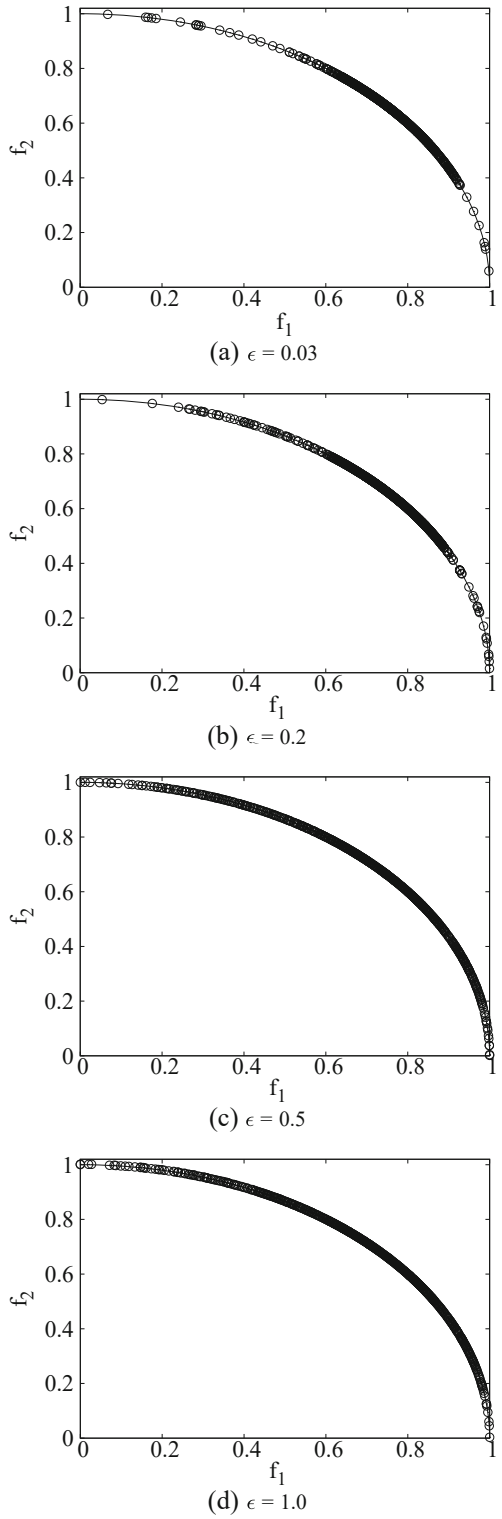


Fig. 8 Influence of the dispersion parameter, ϵ, on the size of the robust region obtained for TP2 (continuous line: Pareto front; points: solutions obtained)

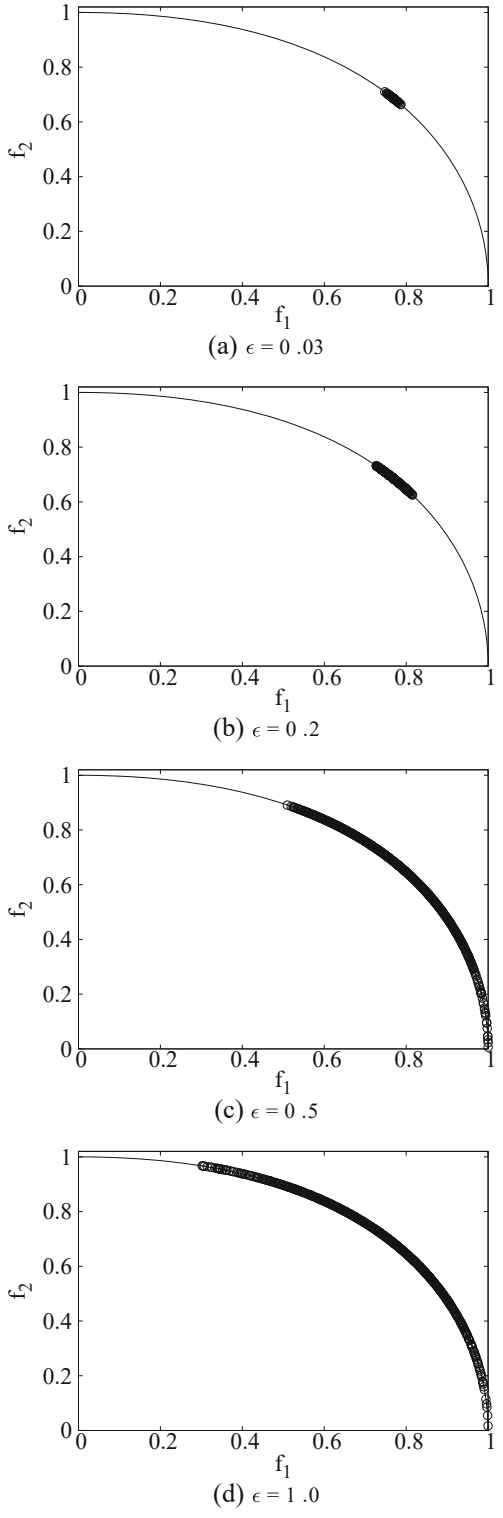


Fig. 9 Influence of the dispersion parameter, ϵ, on the size of the robust region obtained for TP3 (continuous line: Pareto front; points: solutions obtained)

set of robust solutions). With the methodology proposed, this task can be performed automatically by setting the ϵ parameter. In the methods proposed in the literature, this is only possible by a careful trade-off between robustness and

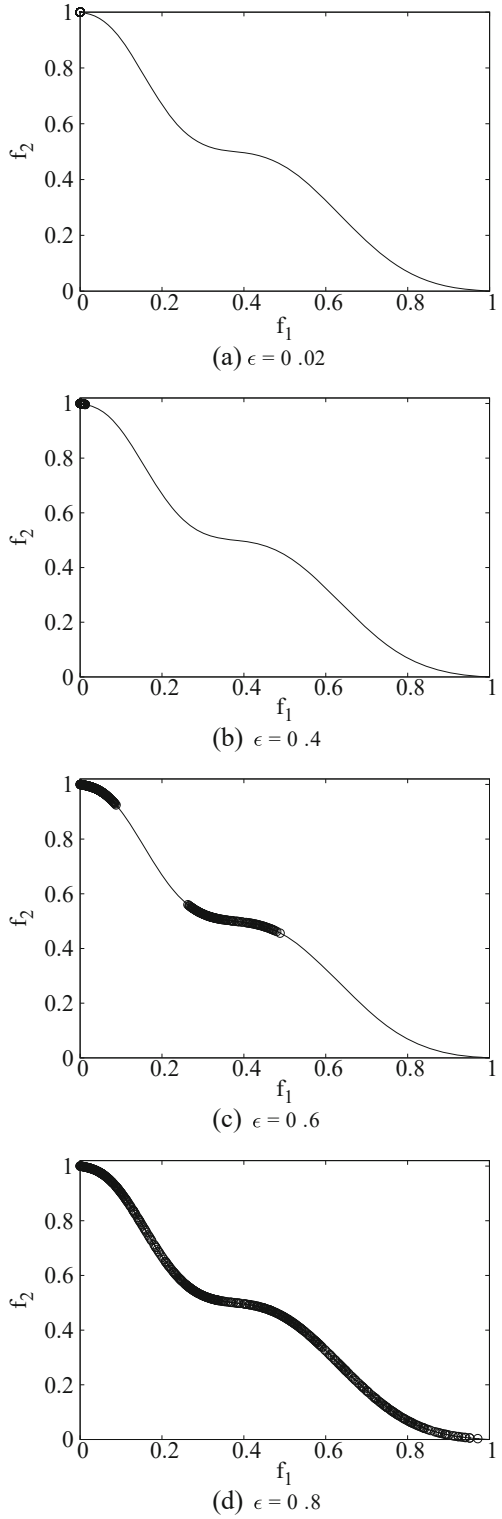


Fig. 10 Influence of the dispersion parameter, ϵ, on the size of the robust region obtained for TP4 (continuous line: Pareto front; points: solutions obtained)

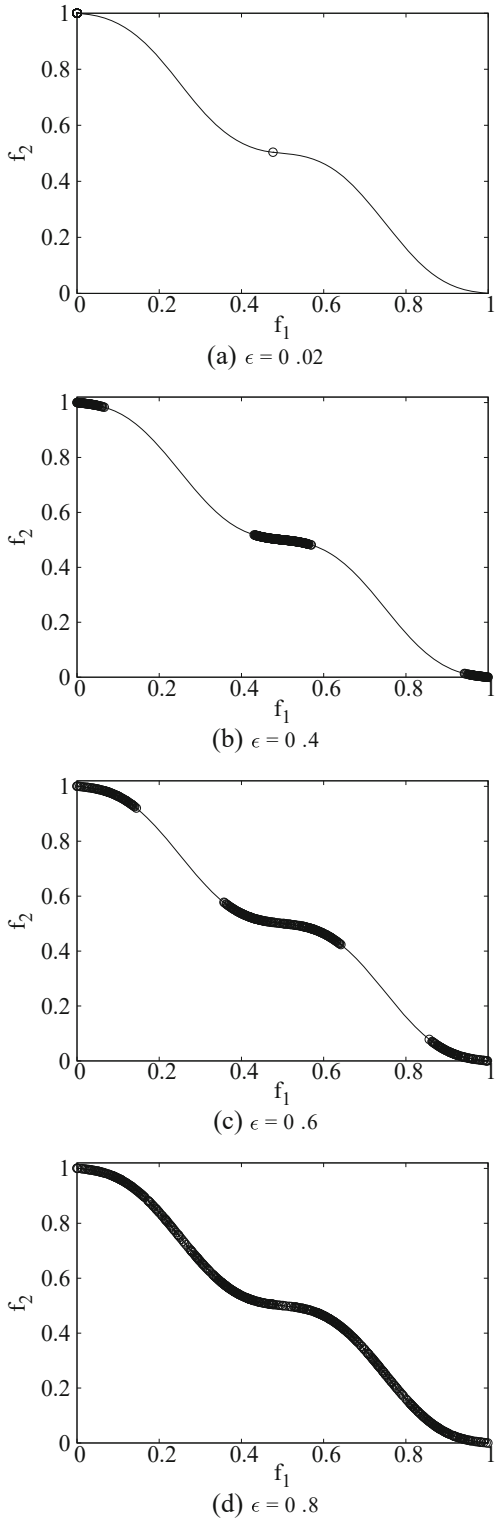


Fig. 11 Influence of the dispersion parameter, ϵ, on the size of the robust region obtained for TP5 (continuous line: Pareto front; points: solutions obtained)

performance after obtaining a set of solutions with varying levels of robustness Li et al. (2005a). Another important advantage is the possibility of adapting this methodology to work with any other multi-objective algorithm, with adding

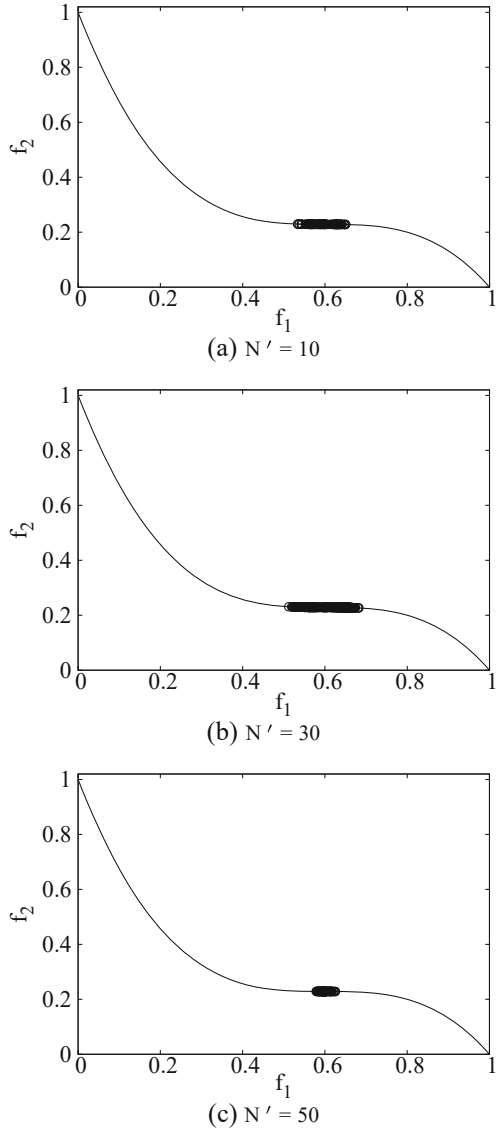


Fig. 12 Influence of number of neighbors, N', for test problem TP1 (continuous line: Pareto front; points: solutions obtained)

the steps identified in Algorithm 2 being the only requirement. This was tested with NSGA-II algorithm, and the results obtained were very similar.

Table 3 Influence of parameters on the algorithm performance

Problem	ϵ	σ_{sh}	Δp	N'	Type	Seed
TP1	very good	no	small	small	x_1	no
TP2	good	0.1	0.1	30/50	x_1	no
TP3	very good	small	small	small	x_1	no
TP4	very good	no	no	no	x_1	no
TP5	very good	no	no	30/50	x_1	no

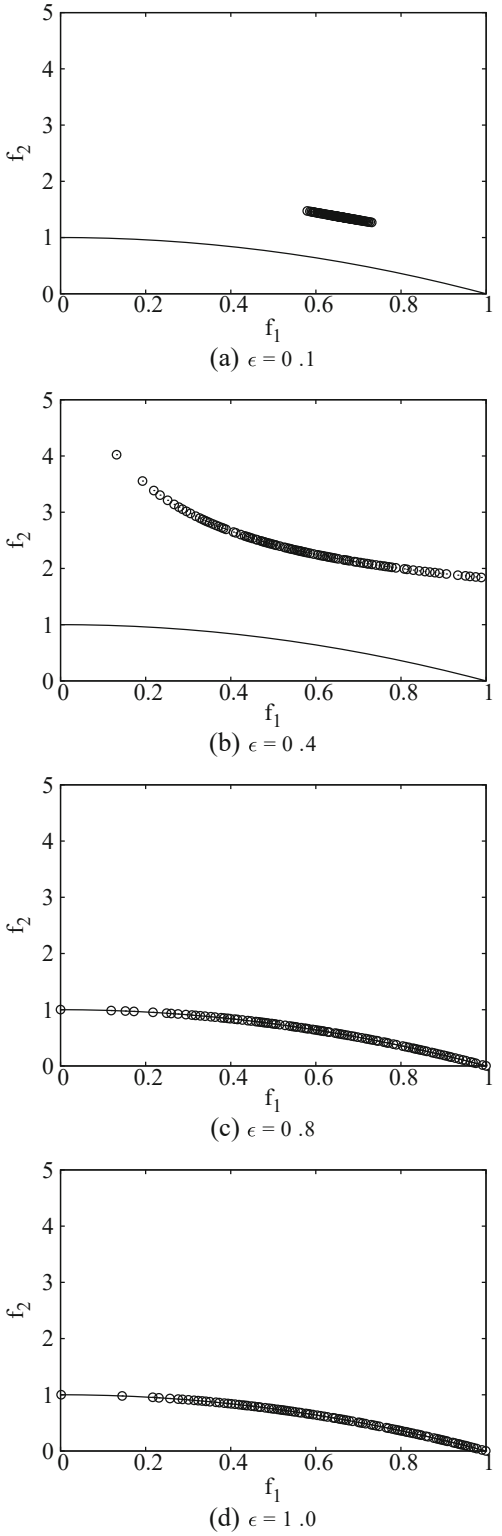


Fig. 13 Influence of the dispersion parameter, ϵ, on the performance of TP6 for Case 1 (continuous line: Pareto front; points: solutions obtained)

Table 4 Computational runs for the comparison of results for TP6 to TP9

Case Study	ϵ	Δp	Type
Case 1	0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0	0.1	x_1
Case 2	0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0	0.05	x_1
Case 3	0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0	0.01	x_1
Case 4	0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0	0.01	–a
Case 5	0.5	0.007, 0.008, 0.009, 0.01, 0.05, 0.07, 0.1	x_1
Case 6	0.3	0.007, 0.008, 0.009, 0.01, 0.05, 0.07, 0.1	–a

[a] two different types of neighbor sampling are used: x_1 and Latin hypercube ($\sigma_{sh} = 0.1$, $N' = 30$ and seed=600)

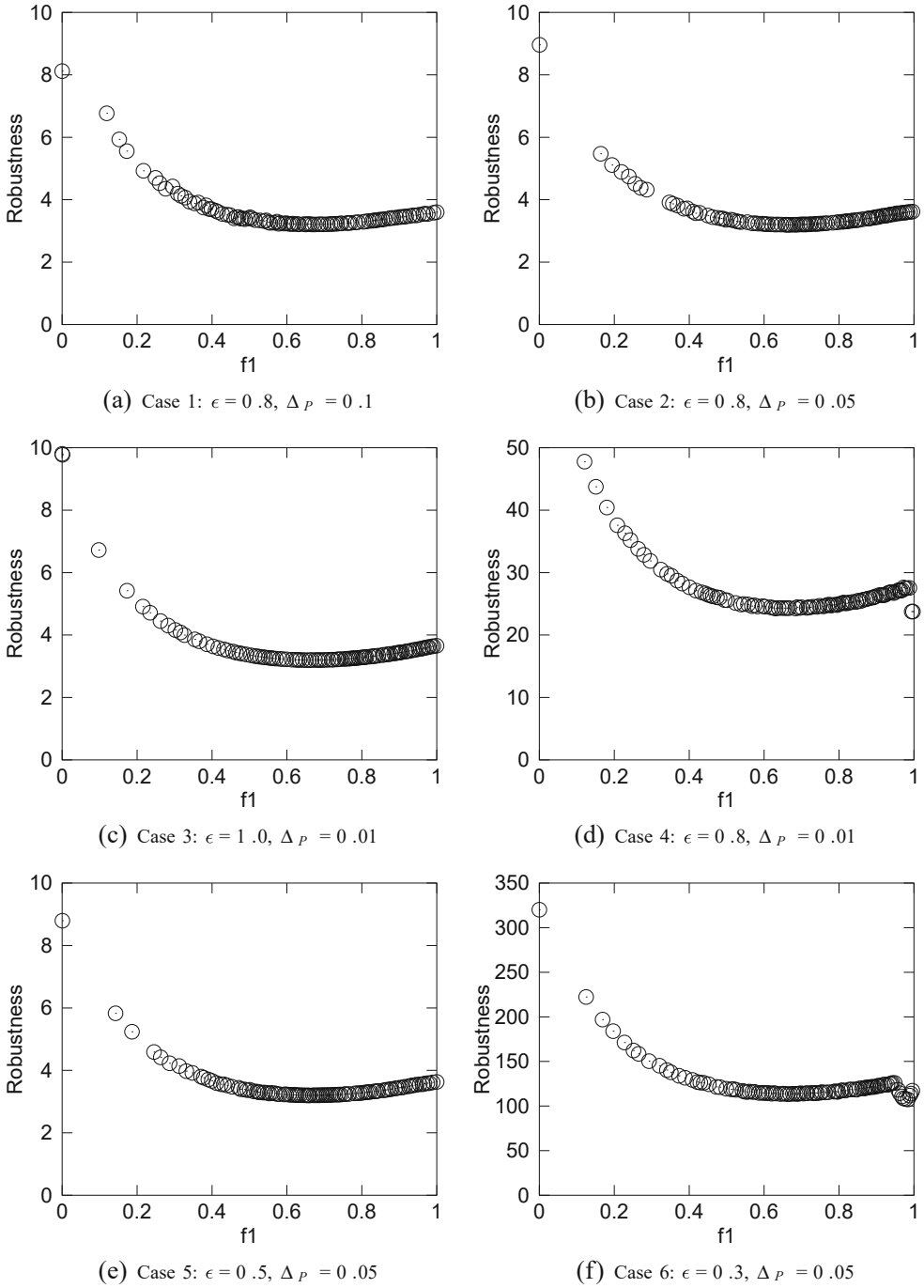


Fig. 14 Robustness for TP6 for all cases presented in Table 4, where ϵ represents the dispersion parameter and Δ_p the minimum distance between neighboring points

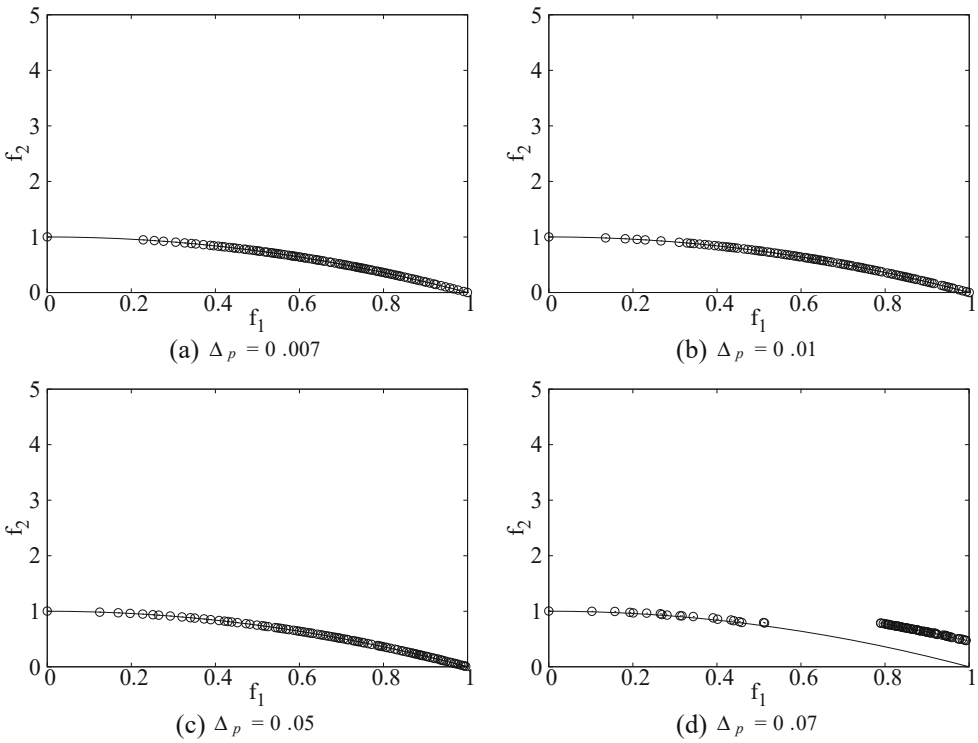


Fig. 15 Influence of the minimum distance between neighboring points, Δ_p, on the performance of TP6 for Case 6 (continuous line: Pareto front; points: solutions obtained)

5.3 Comparison of methods

Table 4 shows the six case studies performed to compare the results for test problems TP6 to TP9. The comparison will be made against the results obtained by Deb and Gupta (2006), from which these test problems were obtained. As shown in Table 4, the study will only consider changes in three different parameters (i.e., ϵ, Δ_p and the type of neighborhood sampling), while the remaining parameters (i.e., σ_{sh}, N' and seed) will be kept constant.

Figure 13 shows the influence of ϵ for case study 1. A complete analysis of all sets of ϵ values tested (see the above web page with supplementary results) shows that the Pareto front is only obtained for $\epsilon \in [0.5, 1.0]$. Similar results were obtained by Deb and Gupta (2006), who used a robustness measure of Type II, as defined in this paper (Figure 31 in reference Deb and Gupta (2006), where TP6 is identified as problem 1). Figure 14 shows the robustness for TP6 as a function of f_1 for all six cases studied here (lower value implies higher robustness). As can be seen, the robustness is larger for values of f_1 greater than 0.2, which is in agreement with the results obtained in Fig. 13. The influence of changes in Δ_p (case study 5) is shown in Fig. 15. In this case, the best values for Δ_p range between 0.007 and 0.05.

Figures 16 to 18 show some of the results (ϵ equal to 0.1 and 0.3) obtained for case study 1 and for test problems TP7, TP8 and TP9. For TP7 (Fig. 16), the proposed method is able to find similar solutions to those in Deb and Gupta (2006) (see Figure 34 in this reference, where TP7 is identified as problem 2). In the case of TP8 (Fig. 17), the proposed methodology is able to converge to the global

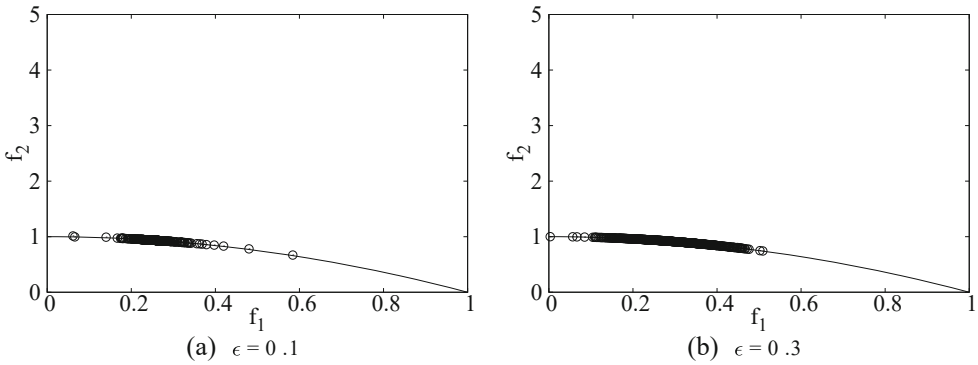
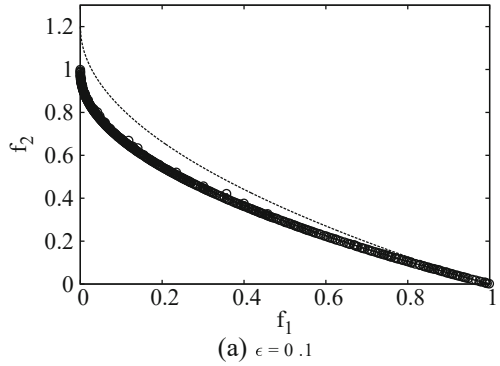


Fig. 16 Influence of the dispersion parameter, ϵ, on the performance of TP7 for Case 1 (continuous line: Pareto front; points: solutions obtained)



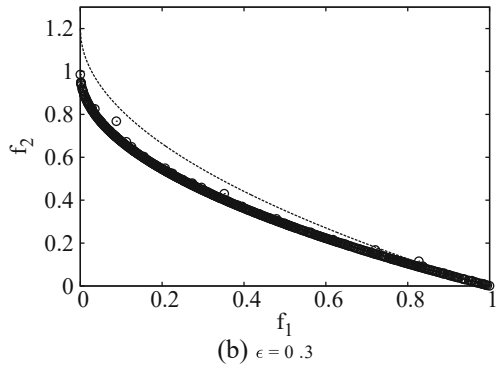
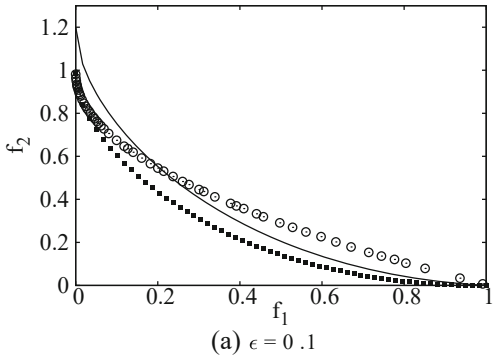


Fig. 17 Influence of the dispersion parameter, ϵ, on the performance of TP8 for Case 1 (continuous line: global Pareto front; discontinuous line: local Pareto front; points: solutions obtained)



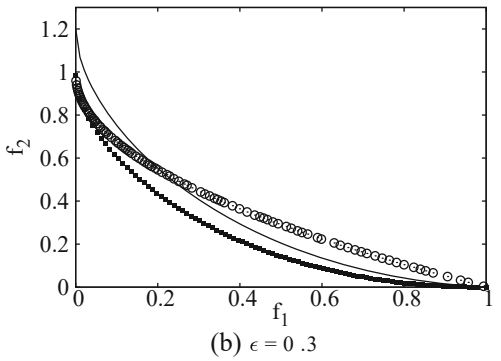


Fig. 18 Influence of the dispersion parameter, ϵ, on the performance of TP9 for Case 1 (continuous line: robust Pareto front; discontinuous line: Pareto front; points: solutions obtained)

front, as shown in Deb and Gupta (2006) (Figure 36, identified as problem 3). Finally, in the case of TP9 (Fig. 18) the algorithm is able to converge partially to the global front ($f_1 < 0.05$). From this point forward (i.e., $f_1 > 0.05$), the convergence depends strongly on the existence of some local fronts. Thus, due to the equation used in the present methodology (10) the robust Pareto front found represents an average of those existing local fronts. These results are not comparable with those found in Deb and Gupta (2006) because in their paper only one of the fronts is found in each run.

A summary of the best values for the parameters studied is shown in Table 5. The comparison of cases 1, 2 and 3 leads to the conclusion that the best solutions are obtained for case 1 ($\Delta_p = 0.1$), as in this case the best Pareto front is obtained for a wide range of ϵ values. Additionally, using the same comparison strategy, the use of the Latin Hypercube for sampling the neighbors (Cases 4 and 5) is not a good option in this set of test problems, as these problems were developed to study robustness for changes in x_1. These results show that the proposed methodology is able to obtain solutions comparable with the method proposed by Deb and Gupta (2006) using a variation of the NSGA-II. The main advantages of the proposed methodology are the following: it is able to obtain these results using a more flexible scheme because it works well on problems with different characteristics (TP1 to TP5), and it enables the possibility of using

Table 5 Computational runs for comparison of results for TP6 to TP9

Test problems	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
TP6	[0.5, 1.0] [a]	[0.5, 0.8]	1.0	[0.3, 1.0]	0.007, 0.008 0.05, 0.07, 0.1	[0.007, 0.05]
TP7	0.1, 0.3	0.3	0.4, 0.8[a]	0.5, 0.8, 0.1[b]	–d	0.007
TP8	[0.1, 1.0]	[0.1, 1.0]	[0.1, 1]	[0.8, 1.0][c]	[0.007, 0.1]	–d
TP9	[0.1, 1.0]	[0.1, 1.0]	[0.2, 1.0]	[0.07, 0.1]	[0.007, 0.1]	0.1

[a] the use of [] means that the best parameter values range between the values identified; [b]: the algorithm converges to the entire front; [c]: for $\epsilon = 0.5$, the mean global front is obtained; [d]: the algorithm does not converge in any of the cases

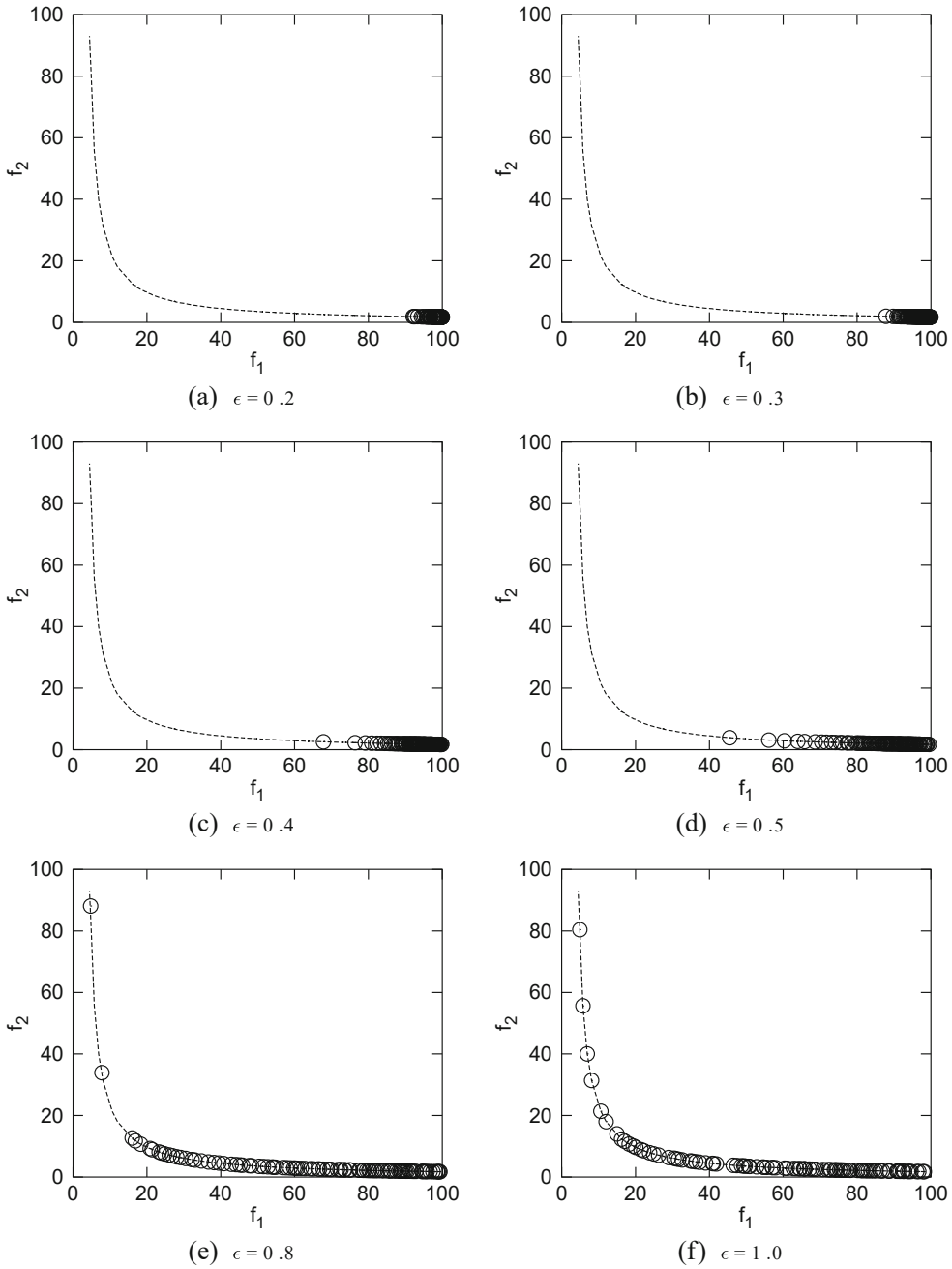


Fig. 19 Influence of the dispersion parameter, ϵ, on the size of the robust region obtained for TP10 (Case 1)

parameters that can be adapted to the problem under study (e.g., ϵ' and the type of neighbor sampling).

5.4 Application to a real engineering problem

The real engineering problem, identified as TP10, was optimized using the following parameter values: $\epsilon = 0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0$, $\Delta_p = 0.1$, $N' = 30$ and $\sigma_{sh} = 0.1$. Four different cases were studied using different strategies for sampling the neighbors: case 1) direction x_1; case 2) Latin hypercube; case 3) random; and case 4) random proportional to the range between the maximum and minimum values for each variable. Figure 19 shows the results obtained for all the ϵ values tested for case 1. Except for the run where $\epsilon = 0.1$, the algorithm converges to the most robust region, and the size of the Pareto front obtained increases with ϵ, as desired. Figure 20 shows the robustness values for case 1, where it can be seen that the solutions found are located in the regions with higher robustness. The results obtained for the other cases are very similar to that of Fig. 19, except when the Latin hypercube sampling method is used.

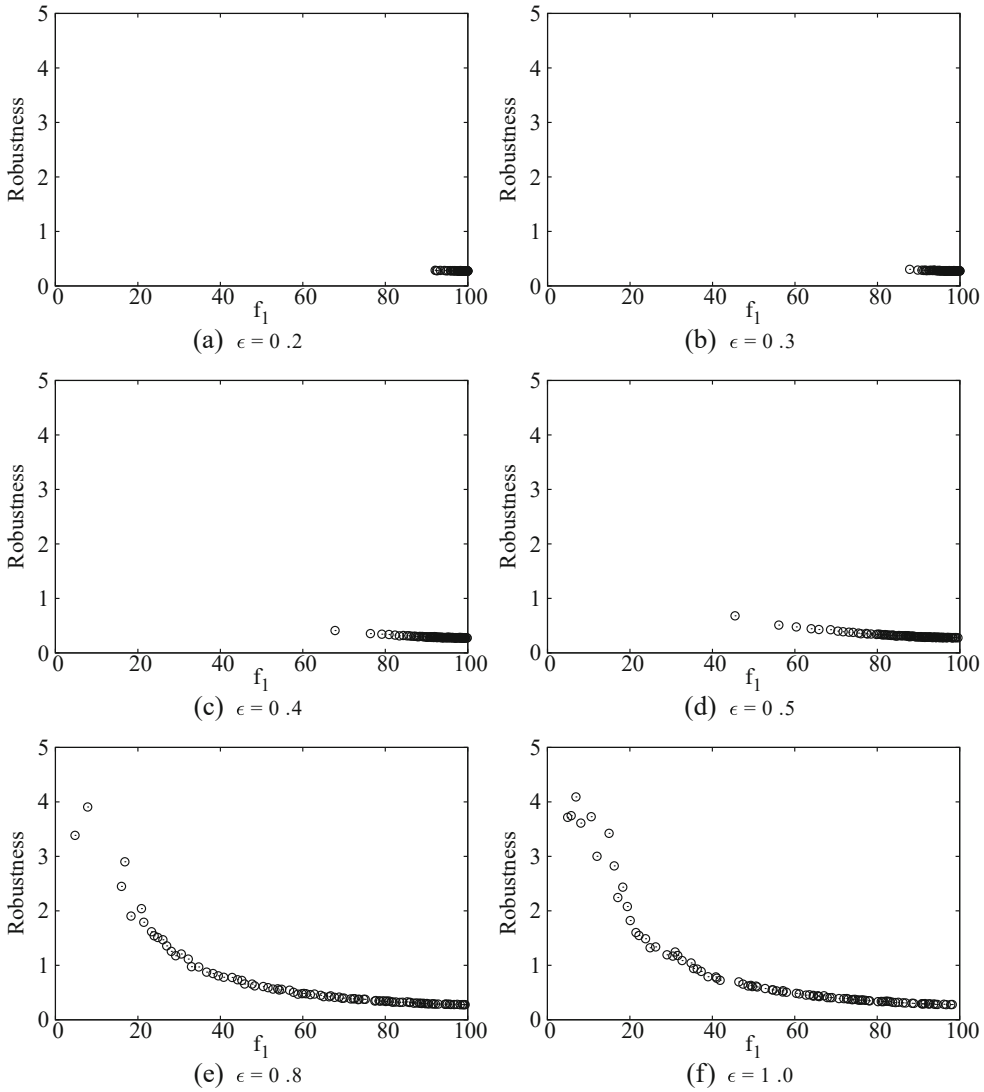


Fig. 20 Influence of the dispersion parameter, ϵ, on the robustness values for TP10 (Case 1)

6 Conclusions

A new approach to robustness analysis in multi-objective optimization problems was proposed. This new approach aimed to obtain the most robust Pareto front solutions and to distribute the solutions along the most robust regions of the optimal Pareto set. A complete study determining the influence of the algorithm parameters was carried out. The studied parameters were the dispersion parameter (ϵ), the maximum distance between solutions (σ_{share}), the minimum distance between the neighbor points selected (Δ_p), the number of neighbors (N'), the type of method used to calculate robustness, and the seed value for random number generation. Further studies were carried out to compare the performance of the proposed method against that of existing methods over five benchmark test problems present in the literature.

A set of test problems accounting for the different types of robustness cases studied in this research was proposed here and used as a source of verification. Several benchmark test problems for robustness studies present in the literature were also used. The method proposed performed as expected for all test problems studied. The resulting Pareto fronts converged to solutions similar to those that can be found in the literature for certain parameter values. Additionally, by fine tuning the algorithm parameters the solutions can be distributed along the most robust regions of the optimal Pareto front.

The proposed methodology can be easily adapted to any other multi-objective optimization algorithm, as it is only necessary to change the way fitness is calculated and to add a routine that can sample the neighborhood and compute robustness, i.e., the main structure of the algorithm remains the same.

To make a critical appraisal, the limitations of the proposed approach must be noted. The main limitation of this work lies in the re-sampling nature of the robustness computations. The necessity of calculating the robustness using a sampling technique around the neighborhood of each solution requires additional evaluations. However, as can be seen in Table 3, in most of the problems tested the number of neighbors is smaller than 30. In the other methods available in the literature the evaluation of the solutions in the neighborhood is also necessary. Further efforts must be made in studying a way of reducing the computational load of robustness approaches. Another limitation arises from the use of a large set of parameters. From Table 3, it is possible to conclude that most of the parameters of the method proposed here have little influence on the algorithm's performance. However, special care must be taken in setting up the appropriate parameter values.

Acknowledgments This work was partially supported by the Portuguese Foundation for Science and Technology under grant PEst-C/CTM/LA0025/2011 (Strategic Project - LA 25 - 2011-2012 and by the Spanish Ministerio de Ciencia e Innovación, under the project Gestión de movilidad efficiente y sostenible, MOVES with grant reference TIN2011-28336.

References

Barrico C, Antunes CH, Pires DF (2009) Robustness analysis in evolutionary multi-objective optimization applied to var planning in electrical distribution networks. In: Lecture notes in computer science, vol 5482. Springer, pp 216–227

Bentley P (1999) Evolutionary design by computers. Morgan Kaufmann, San Francisco

Besharati B, Luo L, Azarm S, PK K (2005) Multi-objective single product robust optimization: An integrated design and marketing approach. J Mech Des 128(4):884–892

Beyer HG, Sendhoff B (2007) Robust optimization: a comprehensive survey. Comput Methods Appl Mech Eng 3190–3218

Branke J (1998) Creating robust solutions by means of an evolutionary algorithm. Parallel Probl Solv Nat 119–128

Branke J (2000) Efficient evolutionary algorithms for searching robust solutions. ACDM 275–286

Branke J, Deb K, Miettinen K, Slowinski R (2008) Multiobjective optimization: interactive and evolutionary approaches, LNCS. Springer, Berlin, Heidelberg

Coello CA, Lamont GB, Veldhuizen DAV (2006) Evolutionary algorithms for solving multi-objective problems (Genetic and evolutionary computation). Springer-Verlag New York, Inc., Secaucus

Costa L, Oliveira P (2007) Dimension reduction in multiobjective optimization. PAMM 7(1):2060,047–2060,048

Damen A, Weiland S (2002) Robust control. Tech. Rep. 5P430, Eindhoven, The Netherlands: Dept. Elect. Eng., Univ. Technol.

Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley, New York

Deb K, Goldberg DE (1989) An investigation of niche and species formation in genetic function optimization. In: Proceedings of the third international conference on genetic algorithms. Morgan Kaufmann, pp 42–50

Deb K, Gupta H (2006) Introducing robustness in multi-objective optimization. Evol Comput 14(4):463–494

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Ferreira J, Fonseca C, Gaspar-Cunha A (2007) Selection of solutions in a multi-objective environment: polymer extrusion a case study. In: Evolutionary methods for design, optimization and control. CIMNE, Barcelona, pp 197–202

Ferreira J, Fonseca CM, Covas JA, Gaspar-Cunha A (2008) Evolutionary multi-objective robust optimization. In: Kosinski W (ed) Advances in evolutionary algorithms, pp 261–278

Fonseca CM, Fleming PJ (1993) Genetic algorithms for multiobjective optimization: formulation, discussion and generalization

Gaspar-Cunha A (2009) Modeling and optimization of single screw extrusion. Lambert Academic Publishing, Koln

Gaspar-Cunha A, Covas J (2004) RPSGAe – reduced Pareto set genetic algorithm: application to polymer extrusion. In: Metaheuristics for multiobjective optimisation. Springer, Berlin, Heidelberg, pp 221–249

Gaspar-Cunha A, Covas J (2008) Robustness in multi-objective optimization using evolutionary algorithms. Comput Optim Appl 39:75–96

Gaspar-Cunha A, Vieira A (2004) A hybrid multi-objective evolutionary algorithm using an inverse neural network. In: Hybrid Metaheuristics (HM 2004). Workshop at ECAI, Valencia, pp 25–30

Goldberg D, Richardson J (1987) Genetic algorithms with sharing for multimodal function optimization. In: Proceedings of second international conference on genetic algorithms. Morgan Kaufmann, pp 41–49

Gunawan S, Azarm S (2005) Multi-objective robust optimization using a sensitivity region concept. Struct Multidiscip Optim 29:50–60

Horn J, Nafpliotis N, Goldberg DE (1994) A niched Pareto genetic algorithm for multiobjective optimization. In: Proceedings of the first IEEE conference on evolutionary computation, IEEE world congress on computational intelligence, pp 82–87

Hu W, Azarm S, Almansoori A (2011) Multi-objective robust optimization under interval uncertainty using online approximation and constraint cuts. J Mech Des 133(6):061,002–061,002–9

Hu W, Azarm S, Almansoori A, Li M (2012) New Approximation Assisted Multi-objective collaborative Robust Optimization (new AA-McRO) under interval uncertainty. Struct Multidiscip Optim 47(1):19–35

Jin Y, Branke J (2005) Evolutionary optimization in uncertain environments-a survey. IEEE Transactions on Evolutionary Computation 9(3):303–317

Jin Y, Sendhoff B (2003) Trade-off between performance and robustness: An evolutionary multiobjective approach. In: Proceedings of second international conference on evolutionary multi-criteria optimization. LNCS 2632, Springer, pp 237–251

Jin Y, Olhofer M, Sendhof B (2002) IEEE Trans Evol Comput 6:481–494

Kirsch U (1981) Optimal structural design. McGraw-Hill, New York

Knowles J, Corne D (2003) Properties of an adaptive archiving algorithm for storing nondominated vectors. IEEE Trans Evol Comput 7(2):100–116

Knowles JD, Corne DW (2000) Approximating the nondominated front using the Pareto archived evolution strategy. Evol Comput 8(2):149–172

Lee J, Kwon YS (2013) Conservative multi-objective optimization considering design robustness and tolerance: a quality engineering design approach. Struct Multidiscip Optim 259–272

Li M, Azarm S, Aute V (2005a) A multi-objective genetic algorithm for robust design optimization. Mech Eng 771–778

Li M, Boyars A, Azarm S (2005b) A new deterministic approach using sensitivity region measures for multi-objective robust and feasibility robust design optimization. J Mech Des 128(4):874–883

Li M, Azarm S, Williams N, Al Hashimi S, Almansoori A, Al Qasas N (2009) Integrated multi-objective robust optimization and sensitivity analysis with irreducible and reducible interval uncertainty. Eng Optim 41(10):889–908

Montgomery DC (2001) Design and analysis of experiments. Wiley, New York

Ray T (2002) Constrained robust optimal design using a multiobjective evolutionary algorithm. In: Proceedings of the 2002 congress on evolutionary computation, vol 1, pp 419 –424

Saha A, Ray T (2011) Practical robust design optimization using evolutionary algorithms. J Mech Des 13:101012

Srinivas N, Deb K (1994) Multiobjective optimization using non-dominated sorting in genetic algorithms. Evol Comput 2:221–248

Tsutsui S, Ghosh A (1997) Genetic algorithms with a robust solution searching scheme. IEEE Trans Evol Comput 1(3):201–208

Wiesmann D, Hammel U, Bäck T (1998) Robust design of multilayer optical coatings by means of evolution strategies. IEEE Trans Evol Comput 2:162–167

Žiha K (2000) Redundancy and robustness of systems of events. Probab Eng Mech 15(4):347–357

Zitzler E, Laumanns M, Thiele L (2001) Spea2: Improving the strength Pareto evolutionary algorithm. Tech. Rep. 103. Swiss Federal Institute of Technology, Zurich